Numerical Modeling of Hydrodynamic Turbulence with Self-gravity on Intel Xeon Phi KNL

Igor Kulikov[1]([✉]) [iD], Igor Chernykh[1], Evgeny Berendeev[1], Victor Protasov[1,2], Alexander Serenko[1], Vladimir Prigarin[1,2], Ivan Ulyanichev[1,2], Dmitry Karavaev[1], Eduard Vorobyov[3], and Alexander Tutukov[4]

[1] Institute of Computational Mathematics and Mathematical Geophysics SB RAS, Novosibirsk, Russia
kulikov@ssd.sscc.ru, chernykh@parbz.sscc.ru, evgeny.berendeev@gmail.com,
inc_13@mail.ru, fafnur@yandex.ru, vovkaprigarin@gmail.com,
wmzonacomvn@mail.ru, kda@opg.sscc.ru
[2] Novosibirsk State Technical University, Novosibirsk, Russia
[3] University of Vienna, Vienna, Austria
eduard.vorobiev@univie.ac.at
[4] Institute of Astronomy RAS, Moskva, Russia
atutukov@inasan.ru

Abstract. In this paper, we present the results of numerical simulations of hydrodynamic turbulence with self-gravity, employing the latest Intel Xeon Phi accelerators with KNL architecture. A new vectorized numerical method with a high order of accuracy on a local stencil is described in details. We outline the main features of the program implementation of the method for massively parallel architectures and study the code parallel implementation. We achieved a performance of 173 gigaFLOPS and an acceleration factor of 48 using a single Intel Xeon Phi KNL. Using 16 accelerators, we were able to achieve a scalability of 97%.

Keywords: Computational astrophysics · Intel Xeon Phi · Numerical methods

1 Introduction

The study of physical processes in the Universe, their influence on the self-organization and evolution of astronomical objects, as well as on their further dynamics and interaction constitute the subject of modern astrophysics. The importance of considering gravitational and magnetic fields and the difficulty of reproducing cosmic conditions in the laboratory impose significant restrictions on the experimental study of astronomical objects. Thus, mathematical modeling is the main, and often the only, approach to the theoretical study of astrophysical processes and astronomical objects.

The evolution of hydrodynamic turbulence and the formation of compact objects as a result of gravitational collapse are among the important processes occurring in astrophysical objects at various spatial scales [1,2]. Magnetohydrodynamic (MHD) turbulence was simulated at the scales of clusters of galaxies in [3]. Problems of gravitational and magneto-gravitational instability [4], dynamics of clouds falling into a black hole [5], and cloud collapse and its fragmentation [6] have been considered in the context of modeling the dynamics of molecular clouds.

An important role is given to the influence of magnetic fields on the evolution of interstellar turbulent flows, in which the magnetic fields are quite strong [7–9]. The energy spectrum [10], the subalfvenian flows [11], and the star formation rate [12] have been studied in the context of the evolution of MHD turbulence. A comparison of various codes for simulation of supersonic turbulence was made in [13]. Turbulence in the solar wind was investigated in [14]. It has been noted that turbulence is the main mechanism for the transition of the deflagration process into detonation in supernova explosion problems [15]. It is important to realize that significant computational high-performance resources are required if one wants to simulate the evolution of hydrodynamic turbulence with self-gravity taken into account.

A trend for using hybrid supercomputers equipped with graphics accelerators and Intel Xeon Phi or Sunway accelerators has become obvious. There are a variety of codes adapted for hybrid supercomputers to simulate hydrodynamic flows in astrophysics [16–23]. However, the main potential for improving the performance in hydrodynamic computing on Intel Xeon Phi accelerators using low-level vectorization of computations has not been sufficiently explored.

In this paper, we shall consider the model problem of turbulence evolution using a new vectorized code developed for supercomputers equipped with Intel Xeon Phi KNL accelerators. The peak performance of Intel Xeon Phi dual accelerators is about three teraFLOPS. Of course, such a value is unreachable in real-world applications but a value of the order of one teraFLOPS can be achieved on synthetic tests. We will be guided by this value when designing our computational model. At present, some program codes (based on publications in the Computer Physics Communications journal) using Intel Xeon Phi accelerators have been implemented in the fields of plasma physics [24], molecular dynamics [25,26], statistical mechanics [27], and hydrodynamics [28].

In 2015, we developed the AstroPhi code [18], based on the implementation of an original numerical method by using the offload programming model of the Intel Xeon Phi. The used accelerator architecture did not allow us to implement vector instructions, although switching to the native mode made it possible to achieve a code performance of 28 gigaFLOPS [29]. The use of low-level vectorization of cycles in the AstroPhi code allowed us to increase the performance to a value of the order of 100 gigaFLOPS [30]. There became evident the necessity to use low-level vectorizing tools to achieve a maximum performance. The new version of the code was based on the HLL method and used a single accelerator

[31,32]. With this implementation, we achieved performances of 245 gigaFLOPS on Intel Xeon Phi 7250 and 302 gigaFLOPS on Intel Xeon Phi 7290.

The computational model and the numerical method will be briefly described in Sect. 2. Section 3 is devoted to the development and investigation of the parallel implementation. In Sect. 4, we formulate the main problems of vectorization. Section 5 is devoted to the simulation of hydrodynamic turbulence taking self-gravity into consideration. Finally, we summarize the conclusions of our research in Sect. 6.

2   The Computational Model

The mathematical model is based on the equations of multicomponent gravitational hydrodynamics. An important condition for the subsequent construction of a vectorized numerical method is to write the equations in vector form. We will use an overdetermined system of hydrodynamic equations with an entropy equation. This will enable us to write the system of hydrodynamic equations in a divergent form, making it possible to formulate a vector numerical method:

$$\frac{\partial}{\partial t}\begin{pmatrix} \rho \\ \rho_i \\ \rho\boldsymbol{u} \\ \rho S \\ \rho E \end{pmatrix} + \bigtriangledown \cdot \begin{pmatrix} \rho\boldsymbol{u} \\ \rho_i\boldsymbol{u} \\ \rho\boldsymbol{u}\otimes\boldsymbol{u}+p \\ \rho S\boldsymbol{u} \\ (\rho E+p)\,\boldsymbol{u} \end{pmatrix} = \begin{pmatrix} 0 \\ s_i \\ \rho\bigtriangledown\varPhi \\ (\gamma-1)\,\rho^{1-\gamma}\,(\varLambda-\varGamma) \\ \varLambda-\varGamma \end{pmatrix}, \qquad (1)$$

where ρ_i is the density of the species, $\rho = \sum_i \rho_i$ denotes the density of the gas mixture, $\boldsymbol{u} = (u_x, u_y, u_z)$ is the velocity vector, S stands for the entropy, $p = p\,(\rho, S, T)$ denotes the pressure, γ is the adiabatic index, $\rho E = \rho\varepsilon + \frac{1}{2}\rho\boldsymbol{u}^2$ is the total mechanical energy, T is the temperature, s_I represents the rate of formation of the corresponding species and, finally, $\varPhi$ is the gravitational potential satisfying the Poisson equation

$$\bigtriangleup\varPhi = 4\pi G\rho, \qquad (2)$$

in which G is the gravitational constant, $\varLambda$ is the cooling function and $\varGamma$ is the heating function. In this article, we restrict ourselves to considering the equation of state based on a combination of the isothermal and adiabatic regimes:

$$p = c_s^2\rho + c_s^2\rho_{\mathrm{crit}}\,(\rho/\rho_{\mathrm{crit}})^\gamma, \qquad (3)$$

where c_s^2 is the isothermal velocity of sound and ρ_{crit} is the critical density of the gas during the transition from isothermal to adiabatic mode, which can be expressed as

$$\rho_{\mathrm{crit}} = \mu m_H n_{\mathrm{crit}}, \qquad (4)$$

with μ the average molecular weight of gas, m_H the mass of a hydrogen atom, and n_{crit} the critical gas concentration. In this work, we assume $n_{\mathrm{crit}} = 10^{10}$ cm^{-3}.

We will consider neither cooling/heating processes nor chemical kinetics processes. Consequently, to simulate hydrodynamic turbulence, we will use the following simplified form of the equations:

$$\frac{\partial}{\partial t}\begin{pmatrix} \rho \\ \rho\boldsymbol{u} \end{pmatrix} + \bigtriangledown \cdot \begin{pmatrix} \rho\boldsymbol{u} \\ \rho\boldsymbol{u}\otimes\boldsymbol{u}+p \end{pmatrix} = \begin{pmatrix} 0 \\ \rho\bigtriangledown\varPhi \end{pmatrix}. \tag{5}$$

However, we will describe all the calculations and the structure of the code for the entire system given in (1).

The equations of hydrodynamics can be written in vector form:

$$\frac{\partial U}{\partial t} + \frac{\partial F(U)}{\partial x} = 0. \tag{6}$$

To solve the equations, one can use a numerical method based on a combination of the operator splitting approach, the Godunov method, the HLL scheme, and the piecewise-parabolic method on a local stencil. The flow through the boundary between the left (L) and the right (R) cells is calculated with the help of the equation

$$F = \frac{F\left(-\lambda_{\mathrm{L}}\tau\right)+F\left(\lambda_{\mathrm{R}}\tau\right)}{2} + \frac{c+\|\boldsymbol{u}\|}{2}\left(U\left(-\lambda_{\mathrm{L}}\tau\right)-U\left(\lambda_{\mathrm{R}}\tau\right)\right), \tag{7}$$

where

$$\lambda_{\mathrm{L}} = c-\|\boldsymbol{u}\|, \qquad \lambda_{\mathrm{R}} = c+\|\boldsymbol{u}\|, \tag{8}$$

with $c = \sqrt{\frac{\gamma p}{\rho}}$ the speed of sound. The modification of the parabolas construct given in [33] is based on the reduction of the order of the first element in the parabola.

The application of the procedure suggested in [33] for the construction of a local parabola to increase the order of accuracy would have made more difficult the transition to an adaptive nested mesh, due to the difference in size of the cells. Therefore, we set two features: to take the original PPML approach using a compact template and the ability to integrate parabolas along the characteristics in each cell. To this end, we save the solver notation and, therefore, the parallel computing algorithms. To solve the problems posed, we will rewrite the parabola construction algorithm from [33] and integrate the parabolas within each cell.

The blocks are the parabolas constructed for the numerical scheme. We construct a piecewise-parabolic function $q(x)$ on a regular mesh with step size h on the interval $[x_{i-1/2}, x_{i+1/2}]$. The general equation of the parabola can be written as

$$q(x) = q_i^{\mathrm{L}} + \xi\left(\triangle q_i + q_i^{(6)}(1-\xi)\right),$$

where q_i is the value at the center of the cell, $\xi = (x-x_{i-1/2})h^{-1}$, $\triangle q_i = q_i^{\mathrm{L}}-q_i^{\mathrm{R}}$, and $q_i^{(6)} = 6\left(q_i - 1/2(q_i^{\mathrm{L}} + q_i^{\mathrm{R}})\right)$, according to conservation laws:

$$q_i = h^{-1}\int_{x_{i-1/2}}^{x_{i+1/2}} q(x)\,dx.$$

To construct $q_i^{\mathrm{R}} = q_{i+1}^{\mathrm{L}} = q_{i+1/2}$, we use an interpolation function of second order of accuracy:

$$q_{i+1/2} = 1/2(q_i + q_{i+1}),$$

where $\delta q_i = 1/2(q_{i+1} - q_{i-1})$. The input value for the construction of the parabola is q_i. The output procedure involves all parameters of the parabola on each interval $[x_{i-1/2}, x_{i+1/2}]$.

1. Construct $\delta q_i = 1/2(q_{i+1} - q_{i-1})$ without extreme regularization:

$$\delta_m q_i = \begin{cases} \min(|\delta q_i|, 2|q_{i+1} - q_i|, 2|q_i - q_{i-1}|)\,\mathrm{sgn}(\delta q_i), \\ \qquad\qquad \text{if } (q_{i+1} - q_i)(q_i - q_{i-1}) > 0, \\ 0, \quad \text{if } (q_{i+1} - q_i)(q_i - q_{i-1}) \le 0. \end{cases}$$

2. Compute the boundary values for the parabola:

$$q_i^{\mathrm{R}} = q_{i+1}^{\mathrm{L}} = q_{i+1/2} = 1/2(q_i + q_{i+1}).$$

3. Reconstruct the parabola according to the following equations:

$$\triangle q_i = q_i^{\mathrm{L}} - q_i^{\mathrm{R}}, q_i^{(6)} = 6(q_i - 1/2(q_i^{\mathrm{L}} + q_i^{\mathrm{R}})).$$

To obtain a monotone parabola, we use the following equations for the boundary values $q_i^{\mathrm{L}}, q_i^{\mathrm{R}}$:

$$q_i^{\mathrm{L}} = q_i, \ q_i^{\mathrm{R}} = q_i, \ (q_i^{\mathrm{L}} - q_i)(q_i - q_i^{\mathrm{R}}) \le 0,$$
$$q_i^{\mathrm{L}} = 3q_i - 2q_i^{\mathrm{R}}, \ \triangle q_i q_i^{(6)} > (\triangle q_i)^2,$$
$$q_i^{\mathrm{R}} = 3q_i - 2q_i^{\mathrm{L}}, \ \triangle q_i q_i^{(6)} < -(\triangle q_i)^2.$$

4. Make a final upgrade of the parabola parameters:

$$\triangle q_i = q_i^{\mathrm{L}} - q_i^{\mathrm{R}},$$
$$q_i^{(6)} = 6(q_i - 1/2(q_i^{\mathrm{L}} + q_i^{\mathrm{R}})).$$

At the final stage of the solution of the hydrodynamic equations, we execute an adjustment procedure. In the case of a gas vacuum border, we have

$$\|\boldsymbol{u}\| = \sqrt{2(E - \epsilon)}, \ (E - \boldsymbol{u}^2/2)/E < 10^{-3}. \tag{9}$$

In other regions, we apply an adjustment to ensure a nondecreasing entropy:

$$\rho\epsilon = \left(\rho E - \frac{\rho \boldsymbol{u}^2}{2}\right), \ (E - \boldsymbol{u}^2/2)/E \ge 10^{-3}. \tag{10}$$

This modification provides a detailed balance of energy and ensures a nondecreasing entropy.

After solving the hydrodynamic equations, it is necessary to restore the gravitational potential with respect to the gas density. To this end, we will use a 27-point template to approximate the Poisson equation. The algorithm for solving the Poisson equation consists of three stages:

1. Setting the boundary conditions for the gravitational potential at the boundary of the region.
2. Transforming the density function to the harmonics space. A fast Fourier transform is used for this.
3. Solving the Poisson equation in the harmonics space. Next, it is necessary to perform the inverse fast Fourier transformation of the potential of the harmonics into the functional space of the harmonics.

The details of the method are given in [33].

3   Parallel Implementation

The parallel implementation is based on a multi-level decomposition of the computations:

1. One-dimensional decomposition of the computational domain by means of MPI, which, for consistency with the solution of the Poisson equation, is specified by the FFTW library.
2. One-dimensional decomposition of the computations by means of OpenMP as part of a single process running on a single Intel Xeon Phi accelerator.
3. Vectorization of computations within a single cell.

The geometric decomposition of the computational domain is carried out by means of MPI processes and by means of OpenMP threads. In the case of a decomposition of the computations by means of MPI, it is necessary to take into account overlapping subregions. The compact calculation template allows for the use of only one overlapping layer.

Next, we describe the basic instructions used to implement the method. We will dwell only on the declarative description:

- **_mm512_set1_pd** – Formation of a vector with each element being a scalar.
- **_mm512_load_pd** – Loading the addresses of the eight double elements of the vector.
- **_mm512_mul_pd** – Multiplication of vectors.
- **_mm512_add_pd** – Addition of vectors.
- **_mm512_sub_pd** – Subtraction of vectors.
- **_mm512_stream_pd** – Writing the vector to memory.
- **_mm512_abs_pd** – Getting the absolute value of the vector elements.

The instructions given here are sufficient to implement a numerical method for the solution of the hydrodynamic equations. We used the following line to compile the code:

icc -xMIC-AVX512 -qopenmp -O3 -no-prec-div

-o gooPhi.mic gooPhi.cpp -lm

It is worth noting only the acceleration of the division through the option -no-prec-div, which is recommended when using SSE extensions.

Table 1. Speedup and real performance of the code on a single Intel Xeon Phi

Cores	GFLOPS	Speedup
1	3.63	1.000
2	7.62	2.099
4	14.67	4.041
8	31.49	8.675
16	62.27	17.154
32	109.57	30.184
64	157.66	43.432
72	173.35	47.755
128	131.68	36.275
256	111.19	30.631

We studied the acceleration of the gooPhi code on a 512^3 grid. We measured the time of the numerical method (Total) in seconds on different numbers of logical cores (Cores). The acceleration P (Speedup) was calculated with the formula

$$P = \frac{\text{Total}_1}{\text{Total}_K},\qquad(11)$$

where Total_1 is the computation time on one logical core and Total_K is the computation time on K logical cores. We also assessed the actual performance. Table 1 contains the results on acceleration and performance on a mesh of size 512^3. We achieved a performance of 173 gigaFLOPS and a speedup factor of 48 using a single Intel Xeon Phi KNL.

In addition, we studied the scalability of the gooPhi code on a mesh of size $512 \times 512 \times 512$ points using all logical cores of each accelerator. Thus, each accelerator has a subdomain size of 512^3. For scalability assessment purposes, we measured the time of the numerical method (Total) in seconds while varying the number of Intel Xeon Phi (KNL) accelerators. The scalability T was computed using the formula

$$T = \frac{\text{Total}_1}{\text{Total}_p},\qquad(12)$$

where Total_1 is the computation time for one accelerator when using a single accelerator and Total_p is the computing time for one accelerator when using p accelerators. The results on acceleration are given in Table 2. Using 16 accelerators, we achieved a 97% scalability. Note that this is a fairly high result.

Table 2. Scalability of the code for various numbers of Intel Xeon Phi accelerators

KNL	Scalability
1	1.000
2	0.999
3	0.998
4	0.994
8	0.988
12	0.972
16	0.968

4   Discussion

In this section, we will discuss several important issues related to the organization of computations, constraints, and new features.

1. In the study, we used the eight elements of the vector (four density functions, three components of the velocity, and the entropy). This is connected with the use of all elements of a 512-bit double-precision vector. We hope that the size of the vector in future versions of the processors will be increased. This would allow us to take into account a greater number of species. At the same time, the multiplicity of eight requires in some cases the use of dummy elements for the organization of computations.
2. When writing the first version of the AstroPhi code and performing subsequent studies, an interesting fact emerged: a greater performance is achieved when using separate arrays to describe hydrodynamic quantities (density, angular momentum, pressure, etc.) than when using an array of C/C++ language structures in which each object contains all the information about the cell. Apparently, this is due to the use of a larger cache. This means that, when accessing multiple arrays, the corresponding cache lines are filled. Thus, we efficiently used as many cache lines as arrays. In the case of structures (or 4D arrays as in the present paper), only one or two cache lines were used.
3. In our implementation, we did not use combined instructions of FMA type. Performance tests, especially in linear algebra applications, where the main operation is a daxpy instruction, show that using FMA instructions improves performance. However, this trend was not observed. Moreover, there was a slowdown of the code, after which we decided to reject such instructions.

5   Modeling of Hydrodynamic Turbulence with Self-Gravity

For the simulation, we considered the test problem in the cubic region $[-1; 1]^3$ with $c_s = 0.1$. The initial density was assumed to be 1. The initial velocity perturbations followed a Gaussian distribution [34].

The main analysis of turbulent flows with gravity consists in estimating the Jeans criterion and the free-fall time, during which a local collapse occurs. To estimate of Jeans criterion, let us write the equations of gravitational hydrodynamics in 1D form using the isothermal equation of state:

$$\frac{\partial \rho}{\partial t} + \frac{\partial}{\partial x}(\rho u) = 0,$$
$$\frac{\partial \rho u}{\partial t} + \frac{\partial}{\partial x}(\rho u u) = -\frac{\partial p}{\partial x} - \rho \frac{\partial \Phi}{\partial x},$$
$$\frac{\partial^2 \Phi}{\partial x^2} = 4\pi G \rho,$$
$$p = c_s^2 \rho. \tag{13}$$

The adiabatic term of the equation of state (3) starts working when the critical density is reached. This density is attained during the development of instability. For the analysis, we need the Jeans criterion, which is achieved at the initial stage by using the isothermal equation of state.

We will consider a linear perturbation of the physical variables:

$$\rho = \rho_0 + \rho_1, \quad p = p_0 + p_1, \quad u = u_1, \quad \Phi = \Phi_0 + \Phi_1. \tag{14}$$

Let us rewrite the equations of gravitational hydrodynamics for the considered perturbation of the physical variables:

$$\frac{\partial \rho_1}{\partial t} + \rho_0 \frac{\partial u_1}{\partial x} = 0,$$
$$\frac{\partial u_1}{\partial t} = -\frac{c_s^2}{\rho_0}\frac{\partial \rho_1}{\partial x} - \frac{\partial \Phi_1}{\partial x},$$
$$\frac{\partial^2 \Phi_1}{\partial x^2} = 4\pi G \rho_1. \tag{15}$$

We seek a nontrivial solution proportional to $\exp\left[i\left(kx + \omega t\right)\right]$. Consequently,

$$\frac{\partial}{\partial t} = i\omega, \quad \frac{\partial}{\partial x} = ik.$$

Let us write the equations for (ρ_1, u_1, Φ_1) in the following form:

$$\omega \rho_1 + k\rho_0 u_1 = 0,$$
$$\frac{kc_s^2}{\rho_0}\rho_1 + \omega u_1 + k\Phi_1 = 0,$$
$$4\pi G \rho_1 + k^2 \Phi_1 = 0. \tag{16}$$

By equating to zero the determinant of the system,

$$\begin{vmatrix} \omega & k\rho_0 & 0 \\ \frac{kc_s^2}{\rho_0} & \omega & k \\ 4\pi G & 0 & k^2 \end{vmatrix},$$

we obtain the condition

$$\omega^2 = k^2 c_s^2 - 4\pi G \rho_0. \tag{17}$$

We should write the critical wavenumber of the Jeans criterion in the form

$$k_J = \left(\frac{4\pi G \rho_0}{c_s^2}\right)^{1/2}, \tag{18}$$

and the critical wavelength of the Jeans criterion in the form

$$\lambda_J = \frac{2\pi}{k_J} = \left(\frac{\pi}{G\rho_0}\right)^{1/2} c_s. \tag{19}$$

By applying a perturbation of the wavelength $\lambda > \lambda_J$, we trigger the gravitational instability.

To estimate the free-fall time, we consider the collapse of a homogeneous sphere of mass M and radius R. We need to estimate the time it takes the sphere radius to decrease from R to zero. Let us write the equation for the moment of impulse in the following form:

$$\frac{d^2 r}{dt^2} = -\frac{Gm}{r^2}, \tag{20}$$

where $m = 4\pi \int_0^r r^2 \rho_0 \, dr$ and $M = \frac{4\pi R^3 \rho_0}{3}$. Here we omit the cumbersome but rather trivial computations. It follows from Eq. (20) that

$$dt = -\left(\frac{8\pi G \rho_0}{3}\right)^{-1/2} \left(\frac{r}{R-r}\right)^{1/2} \frac{dr}{R}. \tag{21}$$

By integrating the last equation from the initial state of the sphere $r = R$ to the final stage $r = 0$, when it collapses, we obtain the equation for the free-fall time t_{ff}:

$$t_{\text{ff}} = \left(\frac{3\pi}{32 G \rho_0}\right)^{1/2}. \tag{22}$$

We will use the last equation to find the characteristic time for the local collapse. Obviously, a collapse is not achievable in a hydrodynamic model in that time. However, since the computational cells have finite size we can consider the process of local collapse in various subdomains of the computational domain. That is especially important in the context of the process of star formation and supernovae explosions.

The results of the computational experiments on the evolution of hydrodynamic turbulence are portrayed in Fig. 1. As we can see, density fragmentation occurs throughout the evolution of turbulence. It would be interesting to consider each individual density wave since in the context of star formation these waves can potentially correspond to young stars. It would also be interesting from the point of view of nuclear reactions to consider the high density regions in the case of turbulent combustion of carbon in white dwarfs.
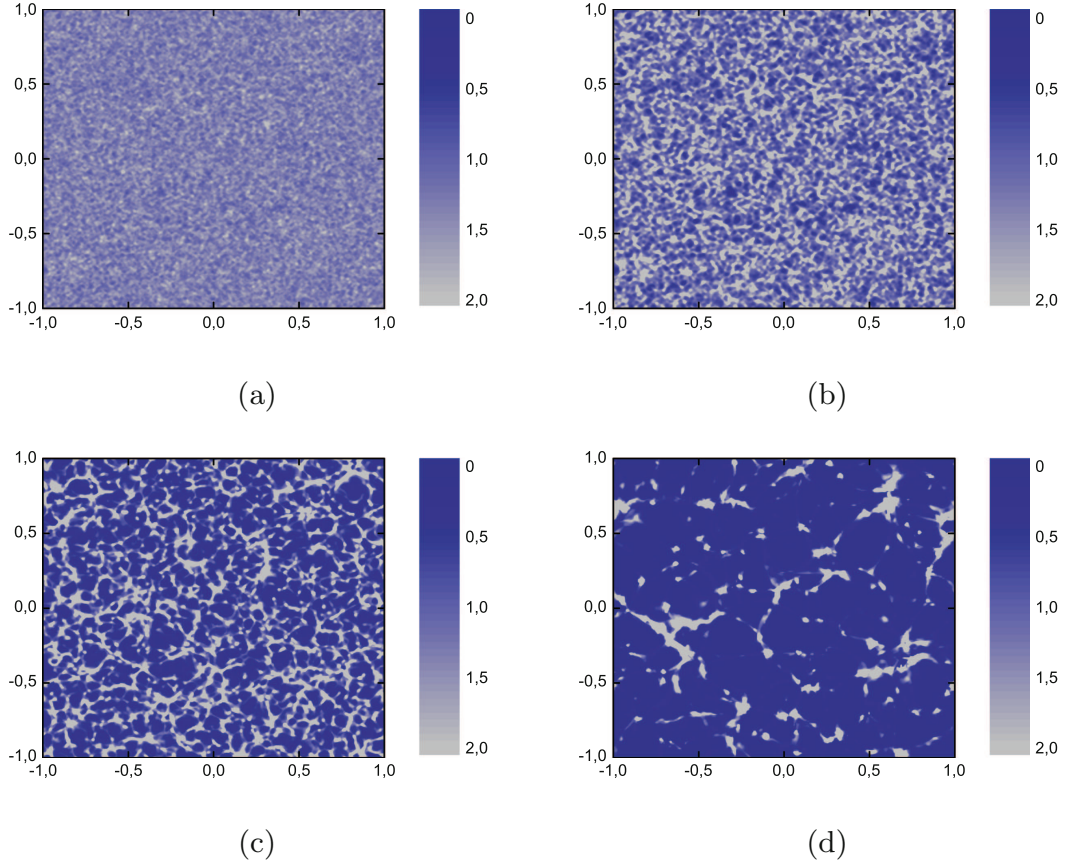
Fig. 1. The density distribution during the evolution of the turbulence process for a model time equal to one quarter (a), two quarters (b), three quarters (c), and four quarters (d) of the free-fall time for cold matter

The problem of hydrodynamic turbulence is one of interest in various astrophysical applications. Our main interest is related to the organization of parallel and distributed computations of supernova explosions. Despite the variety of mechanisms involved in supernova explosions, the distributed computations in these problems are used to correctly reproduce the nuclear combustion of chemical elements and, therefore, correctly compute the injected energy in each computational cell of the domain.

The distributed run of such problems is a very expensive and complicated procedure, and a detailed elaboration is not always required. This is a consequence of the fact that perturbations in the computational cell do not always lead to instabilities. The main criterion for running a hydrodynamic problem should be the analysis of the Jeans criterion λ_J. If it is attained, then it is enough to carry out the simulation for a time less than free-fall time t_{ff}, rather than for the characteristic time step of the main task. All density waves are formed in that time, and this allows one to fully take into account all nuclear reactions in supernovae of all types.

6   Conclusions

In this paper, we presented the results of simulations of hydrodynamic turbulence with self-gravity, employing the latest Intel Xeon Phi accelerators with KNL architecture. A new vector numerical code was described in detail. We achieved a performance of 173 gigaFLOPS and an acceleration factor of 48 by using a single Intel Xeon Phi KNL. Using 16 accelerators, we reached a scalability of 97%.

References

1. Klessen, R., Heitsch, F., Mac Low, M.-M.: Gravitational collapse in turbulent molecular clouds I. Gasdynamical turbulence. Astrophys. J. **535**, 887–906 (2000). https://doi.org/10.1086/308891
2. Heitsch, F., Mac Low, M.-M., Klessen, R.: Gravitational Collapse in turbulent molecular clouds II. Magnetohydrodynamical turbulence. Astrophys. J. **547**, 280–291 (2001). https://doi.org/10.1086/318335
3. Beresnyak, A., Xu, H., Li, H., Schlickeiser, R.: Magnetohydrodynamic turbulence and cosmic-ray reacceleration in galaxy clusters. Astrophys. J. Suppl. Ser. **771**, 131 (2013). https://doi.org/10.1088/0004-637X/771/2/131
4. Kim, W., Ostriker, E.: Amplification, saturation, and Q Thresholds for runaway: growth of self-gravitating structures in models of magnetized galactic gas disks. Astrophys. J. **559**, 70–95 (2001). https://doi.org/10.1086/322330
5. Alig, C., Burkert, A., Johansson, P., Schartmann, M.: Simulations of direct collisions of gas clouds with the central black hole. Mon. Not. Roy. Astron. Soc. **412**(1), 469–486 (2011). https://doi.org/10.1111/j.1365-2966.2010.17915.x
6. Petrov, M., Berczik, P.: Simulation of the gravitational collapse and fragmentation of rotating molecular clouds. Astron. Nachr. **326**(7), 505–513 (2005)
7. Beresnyak, A.: Basic properties of magnetohydrodynamic turbulence in the inertial range. Mon. Not. Roy. Astron. Soc. **422**(4), 3495–3502 (2012). https://doi.org/10.1111/j.1365-2966.2012.20859.x
8. Mason, J., Perez, J.C., Cattaneo, F., Boldyrev, S.: Extended scaling laws in numerical simulations of magnetohydrodynamic turbulence. Astrophys. J. Lett. **735**, L26 (2011). https://doi.org/10.1088/2041-8205/735/2/L26
9. Perez, J.C., Boldyrev, S.: Numerical simulations of imbalanced strong magnetohydrodynamic turbulence. Astrophys. J. Lett. **710**, L63–L66 (2010). https://doi.org/10.1088/2041-8205/710/1/L63
10. Beresnyak, A.: Spectra of strong magnetohydrodynamic turbulence from high-resolution simulations. Astrophys. J. Lett. **784**, L20 (2014). https://doi.org/10.1088/2041-8205/784/2/L20
11. McKee, C.F., Li, P.S., Klein, R.: Sub-alfvenic non-ideal MHD turbulence simulations with ambipolar diffusion II. Comparison with observation, clump properties, and scaling to physical units. Astrophys. J. **720**, 1612–1634 (2010). https://doi.org/10.1088/0004-637X/720/2/1612

12. Federrath, C., Klessen, R.: The star formation rate of turbulent magnetized clouds: comparing theory, simulations, and observations. Astrophys. J. **761**, 156 (2012). https://doi.org/10.1088/0004-637X/761/2/156

13. Kritsuk, A., et al.: Comparing numerical methods for isothermal magnetized supersonic turbulence. Astrophys. J. **737**, 13 (2011). https://doi.org/10.1088/0004-637X/737/1/13

14. Galtier, S., Buchlin, E.: Multiscale hall-magnetohydrodynamic turbulence in the solar wind. Astrophys. J. **656**, 560–566 (2007). https://doi.org/10.1086/510423

15. Willcox, D., Townsley, D., Calder, A., Denissenkov, P., Herwig, F.: Type Ia supernova explosions from hybrid carbon-oxygen-neon white dwarf progenitors. Astrophys. J. **832**, 13 (2016). https://doi.org/10.3847/0004-637X/832/1/13

16. Schive, H., Tsai, Y., Chiueh, T.: GAMER: a GPU-accelerated adaptive-mesh-refinement code for astrophysics. Astrophys. J. **186**, 457–484 (2010). https://doi.org/10.1088/0067-0049/186/2/457

17. Kulikov, I.: GPUPEGAS: a new GPU-accelerated hydrodynamic code for numerical simulations of interacting galaxies. Astrophys. J. Supp. Ser. **214**, 1–12 (2014). https://doi.org/10.1088/0067-0049/214/1/12

18. Kulikov, I.M., Chernykh, I.G., Snytnikov, A.V., Glinskiy, B.M., Tutukov, A.V.: AstroPhi: a code for complex simulation of dynamics of astrophysical objects using hybrid supercomputers. Comput. Phys. Commun. **186**, 71–80 (2015). https://doi.org/10.1016/j.cpc.2014.09.004

19. Schneider, E., Robertson, B.: Cholla: a new massively parallel hydrodynamics code for astrophysical simulation. Astrophys. J. Suppl. Ser. **217**, 2–24 (2015). https://doi.org/10.1088/0067-0049/217/2/24

20. Benitez-Llambay, P., Masset, F.: FARGO3D: a new GPU-oriented MHD code. Astrophys. J. Suppl. Ser. **223**, 1–11 (2016). https://doi.org/10.3847/0067-0049/223/1/11

21. Pekkilaa, J., Vaisalab, M., Kapylac, M., Kapylad, P., Anjum, O.: Methods for compressible fluid simulation on GPUs using high-order finite differences. Comput. Phys. Commun. **217**, 11–22 (2017). https://doi.org/10.1016/j.cpc.2017.03.011

22. Griffiths, M., Fedun, V., Erdelyi, R.: A fast MHD code for gravitationally stratified media using graphical processing units: SMAUG. J. Astrophys. Astron. **36**(1), 197–223 (2015). https://doi.org/10.1007/s12036-015-9328-y

23. Mendygral, P., et al.: WOMBAT: a scalable and high-performance astrophysical magnetohydrodynamics code. Astrophys. J. Suppl. Ser. **228**, 2–23 (2017). https://doi.org/10.3847/1538-4365/aa5b9c

24. Surmin, I., et al.: Particle-in-cell laser-plasma simulation on Xeon Phi coprocessors. Comput. Phys. Commun. **202**, 204–210 (2016). https://doi.org/10.1016/j.cpc.2016.02.004

25. Needham, P., Bhuiyan, A., Walker, R.: Extension of the AMBER molecular dynamics software to Intel's Many Integrated Core (MIC) architecture. Comput. Phys. Commun. **201**, 95–105 (2016). https://doi.org/10.1016/j.cpc.2015.12.025

26. Brown, W.M., Carrillo, J.-M.Y., Gavhane, N., Thakkar, F.M.: Optimizing legacy molecular dynamics software with directive-based offload. Comput. Phys. Commun. **195**, 95–101 (2015). https://doi.org/10.1016/j.cpc.2015.05.004

27. Bernaschia, M., Bissona, M., Salvadore, F.: Multi-Kepler GPU vs. multi-Intel MIC for spin systems simulations. Comput. Phys. Commun. **185**, 2495–2503 (2014). https://doi.org/10.1016/j.cpc.2014.05.026

28. Nishiura, D., Furuichi, M., Sakaguchi, H.: Computational performance of a smoothed particle hydrodynamics simulation for shared-memory parallel computing. Comput. Phys. Commun. **194**, 18–32 (2015). https://doi.org/10.1016/j.cpc.2015.04.006

29. Kulikov, I., Chernykh, I., Tutukov, A.: A new hydrodynamic model for numerical simulation of interacting galaxies on Intel Xeon Phi supercomputers. J. Phys: Conf. Ser. **719**, 012006 (2016). https://doi.org/10.1088/1742-6596/719/1/012006

30. Glinsky, B., Kulikov, I., Chernykh, I., et al.: The co-design of astrophysical code for massively parallel supercomputers. Lect. Notes Comput. Sci. **10049**, 342–353 (2017). https://doi.org/10.1007/978-3-319-49956-7_27

31. Kulikov, I.M., Chernykh, I.G., Glinskiy, B.M., Protasov, V.A.: An efficient optimization of HLL method for the second generation of Intel Xeon Phi processor. Lobachevskii J. Math. **39**(4), 543–550 (2018). https://doi.org/10.1134/S1995080218040091

32. Kulikov, I.M., Chernykh, I.G., Tutukov, A.V.: A new parallel Intel Xeon Phi hydrodynamics code for massively parallel supercomputers. Lobachevskii J. Math. **39**(9), 1207–1216 (2018). https://doi.org/10.1134/S1995080218090135

33. Kulikov, I., Vorobyov, E.: Using the PPML approach for constructing a low-dissipation, operator-splitting scheme for numerical simulations of hydrodynamic flows. J. Comput. Phys. **317**, 318–346 (2016). https://doi.org/10.1016/j.jcp.2016.04.057

34. Kulikov, I., Chernykh, I., Protasov, V.: Mathematical modeling of formation, evolution and interaction of galaxies in cosmological context. J. Phys: Conf. Ser. **722**, 012023 (2016). https://doi.org/10.1088/1742-6596/722/1/012023